REMARKS

In response to the final Office Action mailed on June 7, 2005, Applicant respectfully requests reconsideration of all rejections in the outstanding Office Action in view of the foregoing amendments and following remarks. Claims 1-13, 15, 16, 18, and 19 are currently pending.

I. The Written Description Rejection

Claims 12-20 stand rejected under 35 U.S.C. § 112, first paragraph, as allegedly failing to comply with the written description requirement. Office Action, page 5. Particularly, the Examiner contends that there is no description in the specification about "separately addressable subsets of register objections" as recited in claims 12-20. Moreover, the Examiner contends that there is no description in the specification about "partially redundant expressions" as recited in claims 14, 17, and 20. Applicant respectfully disagrees and traverses the rejection on the following grounds.

The present application as originally filed fully supports claim 12. *See* page 8, paragraph [0128] of the published specification, which clearly specifies that a set of plural register objects are generated, and that a particular variable sized register is represented by a subset of all the register objects in the intermediate representation (emphasis added):

> To avoid conflict between different instructions operating on data of different widths (in this example in a 68000 processor), for each subject processor register the system according to the invention creates a set of three abstract registers, each register of the set being dedicated to data of a given width (i.e. one register for each of byte, word and long word data).

Also, paragraph [0128] shows that the register objects of the subset are separately addressable:

> In the core of a system whose front end is configured to be connected to a 68000, byte values for a subject processor "d0", for example, will be stored in an abstract register labelled "D0_B", whereas word values are stored in a separate abstract register labelled "D0_W", and long values are stored in a third abstract register labelled "D0_L".

Claim 13 is likewise fully supported by the original written description. *See* page 9, paragraph [0136] of the published application referring to Table 1A, which shows situations in which the separately addressable subsets of register objects can concurrently represent the same

variable sized register. In this example, register objects DL_0, DL_W and DO_B concurrently represent the same subject register d0.

Claims 15, 16, 18 and 19 are similarly supported.

Claims 14, 17, and 20 have been cancelled.

Applicant respectfully requests the Examiner to withdraw the written description rejection of claims 12, 13, 15, 16, 18, and 19.

II. The Obviousness Rejection Over Aho In View of Davidson

Claims 1-4, 6, 7, 10, 12-14, and 18-20 stand rejected under 35 U.S.C. § 103(a), as allegedly unpatentable over Aho *et al.*, Compiler, Principles, Techniques, and Tools (1986) ("Aho") in view of U.S. Patent No. 5,613,117 to Davidson *et al.* ("Davidson"). *See* Office Action at page 7. Particularly, the Examiner contends that "Aho teaches all aspects of the applicant's claims but it does not specifically mention 'register objects holding variable values'." *Id.* In an attempt to cure this deficiency, Davidson is introduced as allegedly teaching a "plurality of register objects holding variable values." *Id.* The Examiner then opines that "[i]t would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement the register allocation of Aho with the register holding values further taught by Davidson, for the purpose of allowing the code generator to generate reasonable code for all target machines." *Id.* (citations omitted). Applicant respectfully disagrees and traverses this rejection at least on the following grounds.

As stated in MPEP § 2143.01, to establish *prima facie* obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. *In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974). "All words in a claim must be considered in judging the patentability of that claim against the prior art." *In re Wilson*, 424 F.2d 1382, 165 USPQ 494, 496 (CCPA 1970). If an independent claim is nonobvious under 35 U.S.C. 103, then any claim depending therefrom is nonobvious. *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988).

Applicant respectfully submits that Aho, either taken alone or in combination with Davidson, fails to teach or suggest all of the limitations recited in the claims.

A. Aho Fails To Teach Or Suggest Either Step Recited In Claim 1

The Examiner cites Aho and quotes particularly page 12, final paragraph:

> After syntax and semantic analysis, some compilers generate an explicit intermediate representation of the source program. We can think of this intermediate representation as a program for an abstract machine. This intermediate representation should have two important properties; it should be easy to produce, and easy to translate into the target program.

Aho confirms that intermediate representation is an intermediate step between the source program (which is the starting point as input to the translator or compiler) and the target program (which is the output of the translator or compiler). *See* also Figure 1.9 of Aho and the related discussion on pages 10-11. There are two stages: generating an intermediate representation, and then generating the output target program.

Claim 1 relates to the first stage, namely generating intermediate representation from a source program. Claim 1 is not concerned with the second stage, namely generating target code from the intermediate representation. We refer to the first words of claim 1 (emphasis added):

> A method of generating an intermediate representation of program code...

1. Aho fails to teach or suggest step "(i) generating a plurality of register objects ..."

As identified by the Examiner, Aho does not disclose paragraph (i) of claim 1:

> (i) generating a plurality of register objects holding variable values to be generated by the program code.

This is at least one difference of the claimed invention over the cited prior art.

2. Aho also fails to teach or suggest step "(ii) generating a plurality of expression objects ..."

Aho also does not teach or suggest the step of "generating expression objects..." in "a method of generating intermediate representation," as in claim 1. The Examiner refers to page 49 of Aho. Page 49 is in Chapter 2 concerning "A Simple One-Pass Compiler." Page 49 is irrelevant to producing intermediate representation as in claim 1. The simple syntax tree discussed on page 49 shows how an input string (i.e., in a high level programming language) can be divided into simpler components, and contrasts with a parse tree or concrete syntax tree which retains important semantic information. Syntactical analysis is one of the preliminary stages in a complier prior to producing intermediate representation.

3. Chapter 9 of Aho

Concerning step (ii), the Examiner refers to Chapter 9 of Aho, which discusses "Register Allocation" on page 517 and refers to "Rearranging the Order" on pages 558-559 with reference to Figures 9.18, 9.19 and 9.20. However, the referenced sections on pages 558-9 are irrelevant to the generation of intermediate representation as set forth in claim 1.

Chapter 9 of Aho concerns code generation. Page 513, first paragraph, makes explicitly clear that intermediate representation has already been produced, and that the discussion and explanations in Chapter 9 only concern the generation of target code from the intermediate representation (emphasis added):

> The final phase in our compiler model is the code generator. It takes as input an intermediate representation of the source program and produces as output an equivalent target program, as indicated in Figure 9.1.

Pages 558-9 relied upon by the Examiner are irrelevant to claim 1 and are explicitly taught as irrelevant on page 513. The Examiner is instead referred to Chapter 8 of Aho, which discusses the generation of intermediate representation. However, none of the discussion in Aho relevant to producing intermediate representation, such as Chapter 8, discloses steps (i) and (ii) of claim 1.

4. Item (iii): "wherein at least one variable sized register is represented by plural register objects ..."

The Examiner accepts that Aho does not teach in item (iii) of claim 1 (emphasis added):

> wherein at least one variably sized register is represented by plural register objects, one register object being provided for each possible size of the variable sized register.

That is, Aho does not teach "register objects." Further, Aho does not teach generating intermediate representation by providing plural register objects to represent a variable sized register, when generating intermediate representation of program code.

B. Davidson Fails To Cure Aho's Deficiencies

Applicant respectfully submits that one of ordinary skill in the art would not combine the respective teachings of Aho with Davidson. Nevertheless, such a combination does not arrive at the claimed invention.

Firstly, Davidson does not disclose any of the key features of claim 1. Concerning step (i), the Examiner cites Davidson, column 2, lines 39-44:

> "Next in the internal organisation of a compiler is the register and memory allocation...."

This quotation has to be read in context. Column 2, lines 39-44 refer to processes carried out after intermediate representation has been generated. *See, specifically*, column 2 lines 32-34 (emphasis added):

> After the compiler front end has generated the intermediate language graph and symbol table, various optimising techniques are usually implemented.

Column 2, lines 39-44 of Davidson refer to a step after intermediate representation has been generated, where it is now desired to perform the second stage of the compiler process, namely generating target code.

Next, the Examiner refers to Davidson column 33, line 41-47:

> A data access tuple is a tuple which causes a value to be loaded from or stored into memory. (The word "memory" here includes registers in a register set of the target CPU 25. The only difference between a register and a normal memory location of the CPU 25 is that the "address" of a register can only be used in a data access tuple.) The data access operators are listed in Table 9.

It seems possible that the Examiner has misunderstood the teaching of Davidson. The quoted paragraphs have not "taught us that the tuples may serve as register objects" as alleged by the Examiner. A "tuple" in Davidson is an elemental unit which represents an operation to be performed, such as a load, store, add, label, branch etc. The front end of the compiler creates a data structure with various fields for necessary information associated with each tuple. *See* column 3, lines 21-30:

> The intermediate language representation generated by the front end is based upon a tuple as the elemental unit, where each tuple represents a single operation to be performed, such as a load, a store, an add, a label, a branch, etc. A data structure is created by the front end for each tuple, with fields for various necessary information. Along with the ordered series of tuples, the front end generates a symbol table for all references to variables, routines, labels, etc., as is the usual practice.

Clearly, the tuples of Davidson represent operations, variables and expressions derived from the source program. However, the tuples do not, in generating the intermediate representation, represent register objects as in claim 1.

To quote the Examiner in the Second Office Action, page 8:

> Registers are inherited from microprocessors. In order for Aho and Davidson to use them, registers must be generated at some point.

Applicant respectfully submits that the compiler of Aho compiles from a high-level language down to a machine-oriented language. Similarly, the static compiler taught by Davidson compiles down from a high-level language (such as Pascal or C). The high-level source language is not concerned with registers. Hence, the intermediate representation In Aho and Davidson is not concerned with registers, but is instead concerned with higher-level concepts such as variables ("a", "b", "c") and operations ("*", "+", "-"). Explicit references to registers are generated "at some point," but this point is after intermediate representation has been produced. Register allocation occurs in the second stage of the compiler/translator, when generating target code from the intermediate representation.

Concerning step (ii), the Examiner refers to column 7, lines 43-49 of Davidson. In reply, the tuples of Davidson cannot be "supplemented by the Register Allocation of Aho" as alleged by the Examiner. As discussed above, register allocation is one of the final stages in a compiler relating to code generation, and is irrelevant to generation of intermediate representation.

Concerning item (iii) of claim 1, Applicant respectfully disagrees with the Examiner's interpretation of Davidson. The Examiner refers to column 72, lines 22-25 in Davidson. However, this quotation is completely irrelevant to the claimed invention. Column 72 is an annex concerning "TN Allocation and Lifetime Actions", which are performed by the back end 12 of the disclosed compiler. See the related text at column 25, lines 11-29 (emphasis added):

> A method for doing code generation in the back end 12 by code generator 29...will now be described, referring to Figure 8. ... The TNASSIGN and TNLIFE tasks of the CONTEXT pass use context actions... to analyse the evaluation order to expressions and to allocate TNs with lifetimes nonlocal to the code templates, block 81.

In summary, claim 1 is concerned with "a method of generating intermediate representation...." such as occurs in the front end of a compiler or translator. However, nothing

in either Aho or Davidson teaches the specific items in claim 1 including, in particular, "generating a plurality of register objects...," or "wherein at least one variable sized register is represented by plural register objects, one register object being provided for each possible size of the variable sized register." It is simply impossible for one of ordinary skill in the art to derive the claimed invention from the available prior art.

Similar comments apply to the dependent claims, referring also to the arguments given in response to the First Office Action.

C. Dependent Claims Recite Additional Novel and Nonobvious Limitations

Although dependent claims 2-4, 6, 7 and 12-13 are allowable at least by virtue of their dependency on independent claim 1, these claims recite additional subject matter which is not suggested by the cited art taken either alone or in combination.

For instance, for claim 2 the Examiner refers to Aho page 537 under "Register and Address Descriptors." As discussed above, page 537 is in Chapter 9 of Aho which concerns code generation, and not generation of intermediate representation.

Independent system claim 10 and claims dependent therefrom are allowable at least for the reasons given above.

Claims 14 and 20 have been cancelled.

Applicant respectfully requests the Examiner to withdraw the rejection of claims 1-4, 6, 7, 10, 12-14, and 18-20.

III. <u>The Obviousness Rejection Over Aho In View of Koizumi</u>

Claims 5, 8-9, 11, and 15-17 stand rejected under 35 U.S.C. § 103(a), as allegedly unpatentable over Aho in view of Davidson, and further in view of U.S. Patent No. 5,586,323 to Koizumi *et al.* ("Koizumi"). *See* Office Action at page 17. Particularly with respect to claim 8, the Examiner contends that "Aho teaches all aspects of the applicant's claims but it does not specifically mention 'abstract register.'" *Id.* at page 18. In an attempt to cure this deficiency, Koizumi is introduced as allegedly teaching an "abstract register." *Id.* The Examiner then opines that "[i]t would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement the immediate representation of Aho with the abstract register taught by Kiozumi [*sic*], for the purpose of preserving a form of a program to be executed

repeatedly." *Id.* (citations omitted). Applicant respectfully disagrees and traverses this rejection at least on the following grounds.

Applicant respectfully submits that Aho, either taken alone or in combination with Davidson and/or Koizumi, fails to teach or suggest all of the limitations recited in the claims.

A. Claim 8

Applicant respectfully disagrees with the Examiner's analysis of independent claim 8, for the same reasons as given above. More particularly, claim 8 concerns (emphasis added):

> A method of generating an intermediate representation of program code expressed in terms of the instruction set of a subject processor comprising at least one variable sized register ...

None of the cited prior art, i.e., Aho, Davidson, or Koizumi, discloses such a method where the intermediate representation is generated from source program code which is expressed in terms of the instruction set of a subject processor, and where that subject processor has at least one variable size register. This is simply not mentioned in any of the available prior art documents.

Koizumi does not disclose item (i) of claim 1, namely:

> (i) generating a set of associated abstract register objects representing the variable sized register.

Koizumi refers to "an abstract register machine," which is quite different. Further, none of the other items in claim 1 can be derived from the available prior art, referring to the discussion above.

Independent system claim 11 is allowable for like reasons to those given above.

Claims 5, 9, and 15-17 are allowable at least because they depend on a novel and nonobvious independent claim as discussed above.

Applicant respectfully requests the Examiner to withdraw the rejection of claims 5, 8-9, 11, and 15-17.

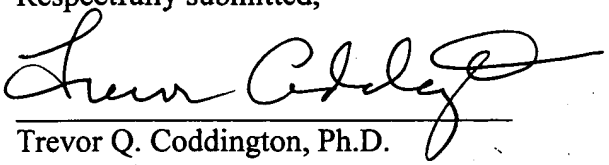
IV. Conclusion

In view of the foregoing, it is respectfully submitted that the present application is in condition for allowance, and an early indication of the same is courteously solicited. The Examiner is respectfully requested to contact the undersigned by telephone at the below listed telephone number, in order to expedite resolution of any issues and to expedite passage of the

present application to issue, if any comments, questions, or suggestions arise in connection with the present application.

Applicant is concurrently filing herewith a Petition for a Two-Month Extension of Time, along with the requisite fee. In the event that a variance exists between the amount tendered and that required by the U.S. Patent and Trademark Office requires to enter and consider this Reply, or to prevent abandonment of the present application, please charge or credit such variance to the undersigned's Deposit Account No. 50-2613 (Order No. 45256.00003.CIP2).

Respectfully submitted,

November 7, 2005 By: 

Trevor Q. Coddington, Ph.D.
Registration No. 46,633

PAUL, HASTINGS, JANOFSKY & WALKER LLP
Customer Number: 36183
P.O. Box 919092
San Diego, CA 92191-9092
Telephone: (858) 720-2500
Facsimile: (858) 720-2555